

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2003-36425

(P2003-36425A)

(43) 公開日 平成15年2月7日(2003.2.7)

(51) Int.Cl. ⁷	識別記号	F I	テマコード*(参考)
G 0 6 K 19/07		B 4 2 D 15/10	5 2 1 2 C 0 0 5
B 4 2 D 15/10	5 2 1	G 0 6 K 17/00	D 5 B 0 3 5
G 0 6 F 9/44		19/00	N 5 B 0 5 8
G 0 6 K 17/00		G 0 6 F 9/06	6 2 0 K 5 B 0 7 6

審査請求 未請求 請求項の数 7 O L (全 15 頁)

(21) 出願番号 特願2001-223775(P2001-223775)

(22) 出願日 平成13年7月25日(2001.7.25)

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 幡野 富久

神奈川県川崎市麻生区王禅寺1099番地 株

式会社日立製作所システム開発研究所内

(72) 発明者 角田 元泰

神奈川県川崎市麻生区王禅寺1099番地 株

式会社日立製作所システム開発研究所内

(74) 代理人 100075096

弁理士 作田 康夫

最終頁に続く

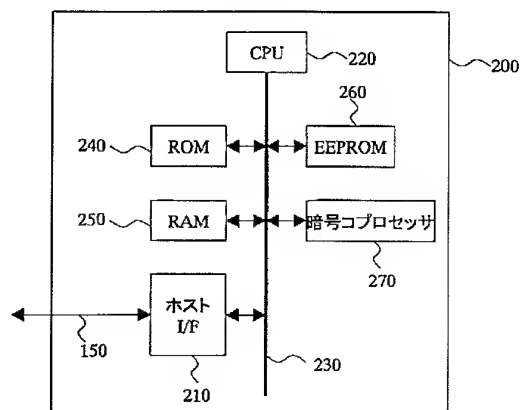
(54) 【発明の名称】 セキュリティカード

(57) 【要約】

【課題】本発明の目的は、複数のアプリケーションを並列に実行することにより、ユーザの利便性を向上することである。

【解決手段】本発明は、ホストインタフェース210と、中央処理装置220と、ROM240と、RAM250と、EEPROM260と、暗号コプロセッサ270とを備え、中央処理装置220が、共通コマンド処理プログラムのロード時に共通コマンド用ワークのサイズを算出し、EEPROM260に記憶させ、アプリケーション専用コマンド処理プログラムのロード時に共通コマンド用ワークエリアをEEPROM260にアプリケーション単位で割付け、論理チャネル単位でアプリケーション専用コマンド処理プログラム(コード、データ)、共有コマンド処理プログラム(コード、ワーク)を対応づけて管理する。

図2



【特許請求の範囲】

【請求項1】ホスト機器からのコマンドに応じて、データの処理及び記憶が可能なセキュリティカードにおいて、

複数のアプリケーション間で共通に使用できる共通コマンド処理プログラムの登録削除コマンドを備えたセキュリティカード。

【請求項2】1つのアプリケーションで使用できるアプリケーション専用コマンド処理プログラムの登録削除コマンドを備えた請求項1に記載のセキュリティカード。

【請求項3】前記アプリケーション専用コマンド処理プログラムの登録削除コマンドで登録したアプリケーションプログラムのデータ部を、前記共通コマンド処理プログラムの登録削除コマンドで登録した共通コマンドのコマンドパラメータとして使用できる請求項2に記載のセキュリティカード。

【請求項4】論理チャネル単位で、前記アプリケーション専用コマンド処理プログラムと前記共有コマンド処理プログラムとを対応づけて管理する手段を備えた請求項2に記載のセキュリティカード。

【請求項5】前記共通コマンド処理プログラムをローディングする場合に共通コマンド命令コードへのポインタおよび共通コマンド用ワークエリアのサイズを算出し記憶装置に記憶させ、前記アプリケーション専用コマンド処理プログラムをローディングする場合に共通コマンド用ワークエリアをアプリケーション単位で前記記憶装置に割付ける手段を備えた請求項2に記載のセキュリティカード。

【請求項6】論理チャネルをオープンする場合に対応するアプリケーション専用コマンド用ダイナミックデータ及び共有コマンド用ワークデータを初期化する手段を備えた請求項2に記載のセキュリティカード。

【請求項7】論理チャネル番号で共通コマンド用ワークエリアを選択し切換える手段、論理チャネル番号でアプリケーション用スタティックデータを選択し切換える手段を備えた請求項2に記載のセキュリティカード。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、セキュリティ機能と記憶機能を搭載し、ホスト機器からのコマンドに応じて、データの処理及び記憶が可能な処理／記憶媒体（例えば、セキュリティカード）、及びその処理／記憶媒体（例えば、セキュリティカード）が挿入可能なホスト機器に関する。

【0002】

【従来の技術】ICカードは、プラスチックカード基板中にIC（集積回路）チップを埋め込んだものであり、その表面にICチップの外部端子を持つ。ICチップの外部端子には電源端子、クロック端子、データ入出力端子などがある。ICチップは、接続装置が外部端子から

電源や駆動クロックを直接供給することによって動作する。ICカードは外部端子を通して端末機などの接続装置との間で電気信号を送受信することにより、接続装置と情報交換をおこなう。情報交換の結果として、ICカードは計算結果や記憶情報の送出、記憶情報の変更をおこなう。ICカードは、これらの動作仕様に基づいて、機密データ保護や個人認証などのセキュリティ処理を実行する機能を持つことができる。ICカードは、クレジット決済やバンキングなど機密情報のセキュリティが必要とされるシステムにおいて、個人識別のためのユーザデバイスとして利用されている。

【0003】

【発明が解決しようとする課題】ICカードは内部に機密データを格納し、機密データに関わる処理は全てICカード内部で実行する。そうすることによって、機密データがICカードの外に出ることを防いでいる。機密データに関わる処理はコマンド処理プログラムとして構成されている。ICカードの外部（ホスト機器）から、ICカードにコマンドを送信すると、そのコマンドに対応したコマンド処理プログラムが実行される。そして実行結果がホスト機器に返信される。コマンドは一つとは限らず複数個備わっている場合が多い、コマンド（あるいはコマンド処理プログラム）は相互に密接な関係があり、前のコマンドの実行結果を内部に保存しておき、そのデータを使って後のコマンドの処理が行われることが多い。予め決められているコマンド発行のシナリオに従って、ホスト機器からコマンドがICカードに発行される。一連のコマンド送信／レスポンス受信を繰り返すことによって、セキュリティシステム処理が遂行される。

【0004】マルチアプリケーション対応のICカード用OS（オペレーティングシステム）としてMULTOSなどが知られている。アプリケーション用コマンド処理プログラムと機密データは一体として管理されていて、アプリケーション単位にICカードに登録・削除できる。そしてアプリケーション間にはファイアウォールが設定されており、相互干渉を防止している。

【0005】上記従来の技術では、アプリケーション間にはファイアウォールが設定されているため、アプリケーション間で共通に使用する共通コマンドを登録できなかった。また、アプリケーション内のデータを共通コマンドへのコマンドパラメータとして使うことなどできなかった。さらに、アクティブなアプリケーションを一つしかオープンできず複数アプリケーションを同時に実行することが難しいという問題点があった。

【0006】本発明の目的は、複数のアプリケーションを並列に実行することにより、ユーザの利便性を向上したセキュリティカードを提供することである。

【0007】

【課題を解決するための手段】本発明は、オープンチャネルコマンド、クローズチャネルコマンド、共通コマン

ド処理プログラムのローディングコマンド、アプリケーション専用コマンド処理プログラムのローディングコマンドを備える。

【0008】また、システムコマンド、共通コマンドおよびアプリケーション専用コマンドを実行する手段、多数の論理チャネルを管理するテーブル、論理チャネル単位にアプリケーション専用コマンド処理プログラム（命令コード、スタティックデータ、ダイナミックデータ）、共有コマンド処理プログラム（命令コード、ワークデータ）を対応づけて管理する手段、共通コマンド処理プログラムをローディングする場合に共通コマンド命令コードへのポインタおよび共通コマンド用ワークエリアのサイズを算出し記憶しておく手段、アプリケーション専用コマンド処理プログラムをローディングする場合に共通コマンド用ワークエリアをアプリケーション単位に割付ける手段、論理チャネルをオープンする場合に対応するアプリケーション専用コマンド用ダイナミックデータと共有コマンド用ワークデータを初期化する手段、論理チャネル番号とコマンド番号でコマンド処理プログラム選択し実行させる手段、論理チャネル番号で共通コマンド用ワークエリアを選択し切換える手段、論理チャネル番号でアプリケーション用スタティックデータを選択し切換える手段、を備える。

【0009】

【発明の実施の形態】本発明の実施例について図面を用いて説明する。

【0010】図1に本発明の一実施例の全体構成図を示す。本発明に関わる統合セキュリティカード200は、ホストインタフェース210を介してホスト機器のアダプタ130に接続する。ホスト機器100は計算機の全体制御を行う中央処理装置（以下CPU）110、メモリ装置120、アダプタ130で構成されていて、それらはシステムバス140で接続されている。メモリ装置120には、アプリケーションプログラム121、122、およびコマンド登録ツール123が格納されている。これらのプログラムは統合セキュリティカード200をアクセスする。プログラム121、122、123から発行されたコマンドは、アダプタ130、インタフェースバス150を経由して統合セキュリティカード200へ送信される。また、統合セキュリティカード200からは、コマンドに対応してレスポンスがホスト機器100へ返信される。

【0011】図2は、本発明に関わる統合セキュリティカード200のシステムブロック図である。

【0012】統合セキュリティカード200は、演算処理を行うためのCPU220と、データ（プログラムを含む）を記憶するためのROM（Read Only Memory）240とRAM（Random Access Memory）250とEEPROM（Electrically Erasable Programmable ROM）260と、暗号/復号に関する処理を行うための暗号コ

ロセッサ270と、外部とデータを送受信するためのホストインタフェース210とを備え、それらは、バス230によって接続される。暗号コプロセッサ270によって、ホスト機器100からのコマンドに応じて、暗号処理を実行することが可能である。尚、暗号コプロセッサ270（ハードウェア）の代わりに、プログラム（ソフトウェア）に従ってCPU220が暗号処理を実行してもよい。統合セキュリティカード200は、ホスト機器100と通信を行い、ホスト機器100からのコマンドに応じて、セキュリティ処理及び記憶処理を実行する。統合セキュリティカード200とホスト機器100との間の通信は、インタフェースバス150のような有線であってもよいし、赤外線や電波のような無線であってもよい。統合セキュリティカード200の形状は、カード形状に限定されない。統合セキュリティカード200は、例えば、ICカードである。

【0013】ホスト機器100から統合セキュリティカード200に発行されるコマンドのタイプを図3に示す。コマンドには、システムコマンド、アプリケーション間で共通に使用できる共通コマンド、アプリケーション専用コマンドの3つのタイプがある。共通コマンドとアプリケーション専用コマンド処理プログラムはホスト機器100から登録することができる。

【0014】図4は、統合セキュリティカード200内のROM240、EEPROM260とRAM250のメモリマップである。ROM240には、コマンド受信プログラム241、コマンド終了プログラム242、システム制御およびシステムコマンド領域243を割り振る。EEPROM260には、システム管理データ領域261、アプリケーション動作領域262、共通コマンド動作領域263を割り振る。RAM250には、システムコマンド用ワーク領域251、アプリケーション用ワーク領域252を割り振る。コマンド受信プログラム241、コマンド終了プログラム242、およびシステムコマンド領域243内のシステムコマンド処理プログラムは、システム管理データ領域261とシステムコマンド用ワーク領域251を使用する。

【0015】図5は、システムコマンド用ワーク領域251と、システム管理データ領域261の詳細である。300はホスト機器100から受取ったコマンド情報を格納しておくコマンドデータ領域、310は逆にホスト機器に返送するレスポンス情報を格納しておくレスポンスデータ領域である。320は現在選択されている論理チャネル番号を記憶するカレントCHCBである。

【0016】また、システム管理データ領域261には、いろいろなシステム管理テーブルを格納する。330は登録されているアプリケーション専用コマンド処理プログラムを管理するテーブルAPCB（Application Control Block）、340は登録されている共通コマンド処理プログラムを管理するテーブルCCCB（Common

command control block)、350は論理チャネルを管理するテーブルCHCB(Channel control block)、360はシステムコマンドを管理するテーブルSCCB(System command control block)である。

【0017】図6は、アプリケーション動作領域262、アプリケーション用ワーク領域252の詳細である。EEPROM260内のアプリケーション動作領域262はアプリケーション単位に分割されていて(510、520)、それぞれの領域に各アプリケーション専用コマンド処理プログラムのコード(511、521)とデータS(512、522)が格納される。データS(512、522)は定数などのスタティックデータである。RAM250内のアプリケーション用ワーク領域252もアプリケーション単位に分割されていて(610、620)、それぞれの領域に各アプリケーション専用コマンド処理プログラムのデータD(611、621)、および共通コマンド用ワーク(612、622)が格納される。データD(611、621)はアプリケーションがアクティブになっている間使用するダイナミックデータである。共通コマンド用ワーク(612、622)は、それぞれのアプリケーション毎に割り振られる。

【0018】図7は、登録されているアプリケーション専用コマンド処理プログラムを管理するテーブルAPCB(330)の詳細である。APCB(330)はアプリケーションに対応してエントリ(331、332、...)を持つ。各エントリ331は対応するアプリケーションに関する情報を格納している。これらの情報はアプリケーションが登録された時に作成される。

【0019】APCB(1,0)はアプリケーションが登録されているかどうかをあらわすスイッチである。対応するアプリケーションが格納されている場所(コード、データS、データDへのポインタ)、アプリケーション名、そのアプリケーションに割り振られた共通コマンド用ワークへのポインタなどがある。APCB(330)は図10に示すアプリケーションユニットのローディングコマンド(Load_AP)で作成される。

【0020】図8は、論理チャネルを管理するテーブルCHCB(350)の詳細である。

【0021】ホスト機器100内のプログラム(121、122、123)は論理チャネルを経由して、統合セキュリティカード200内のアプリケーション専用コマンド、共通コマンドにアクセスできる。論理チャネルは図10に示すオープンチャネルコマンド(Open_Channel)で作成され、論理チャネルとアプリケーションが対応付けられる。論理チャネル番号に対応してエントリ(351、352、...)を持つ。各エントリ(351)は対応する論理チャネルに関する情報を格納している。CHCB(1,0)は当該チャネルを使用中のアプリケーション番号をあらわす。使用中アプリケーションが格納さ

れている場所(コード、データS、データDへのポインタ)、そのアプリケーションに割り振られた共通コマンド用ワークへのポインタを格納する。

【0022】図9は、システムコマンドを管理するテーブルSCCB(360)、および登録されている共通コマンド処理プログラムを管理するテーブルCCCB(340)の詳細である。

【0023】SCCB(360)はシステムコマンド処理ルーチンのアドレスを格納する。CCCB(340)は共通コマンドの登録状態、共通コマンドコードへのポインタ、共通コマンド用ワークのサイズなどを格納する。CCCB(340)は図10に示す共通コマンドのローディングコマンド(Load_CCOM)で作成される。

【0024】図10に、システムコマンドの一覧を示す。Open_Channelコマンドは論理チャネルをオープンし、指定されたアプリケーションを対応付ける、そして当該チャネル番号を返す。Close_Channelコマンドは指定されたチャネルをクローズする。Load_CCOMコマンドは共通コマンド処理プログラムをローディングする。Load_APコマンドはアプリケーションユニットをローディングする。システムコマンドを選択する場合チャネル番号0を使う。

【0025】図11に、Load_CCOMコマンドでロードする共通コマンドのロードユニット700の構成、Load_APコマンドで扱うアプリケーションのロードユニット710の構成を示す。700は共通コマンド属性情報701、共通コマンド命令コード702、共通コマンド用ワーク703で構成される。710は、アプリケーションの属性情報711、アプリケーション命令コード712、スタティックデータS(713)、ダイナミックデータD(714)で構成される。

【0026】図12は、コマンドデータ領域300に格納されるコマンド情報の詳細、レスポンスデータ領域310に格納されるレスポンス情報の詳細である。コマンドデータ300はチャネル番号301、コマンド番号302、コマンドパラメータ303で構成される。システムコマンド発行の場合、チャネル番号は0である。またコマンド番号302でシステムコマンド、共通コマンド、ユーザアプリケーション専用コマンドのいずれであるか区別する。レスポンスデータ310はリターンコード311、コマンドリターンデータ312で構成される。

【0027】図13は統合セキュリティカード200の初期化処理901を示す。統合セキュリティカード200の電源が入ると初期化処理901が動作する。システムコマンド用ワーク領域251を初期化し(ステップ902)、コマンド受信241へ分岐する(ステップ908)。

【0028】図14はコマンド受信処理241、図15はコマンド起動処理1100、図16は無効コマンド処

理1200、コマンド終了処理1300である。コマンド受信処理241では、ホストインタフェース210を監視しコマンドが到着したか否かをチェックする(ステップ1001)。コマンドが到着した場合、コマンドデータをコマンドデータ300に取込む(ステップ1002)。そしてコマンド起動処理1100へ分岐する(ステップ1003)。

【0029】コマンド起動処理1100では、まずチャンネル番号301をチェックする(ステップ1101)。チャンネル番号が0の場合システムコマンドデスパッチ2000へ分岐する(ステップ1108)。

【0030】チャンネル番号が0以外の場合、チャンネル番号に対応したCHCBエントリの番号をカレントCHCB(320)へ設定する(ステップ1102)。以後カレントCHCBに関して制御する。まず、当該チャンネルがオープンしているか否かをチェックする(ステップ1103)。チャンネルがオープンされている場合CHCB(1,0)にチャンネル使用中のアプリケーション番号が設定されている。オープンされていない場合、無効コマンド処理1200へ分岐する(ステップ1120)。オープンされている場合、コマンド番号302が共通コマンドであるか否かをチェックする(ステップ1104)。あるいはアプリケーション専用コマンドであるか否かをチェックする(ステップ1106)。共通コマンドの場合、CCCB(340)をチェックする。CCCB(0)が登録有りの状態ならばCCCB(1)から共通コマンドコードの先頭アドレスを取得する。そして取得したアドレスへ分岐する(ステップ1105、1109、1130)。ここでは、図22の共通コマンドコードの先頭3000へ分岐する。

【0031】一方、アプリケーション専用コマンドの場合、CHCB(1,1)をチェックする。CHCB(1,1)に設定されているアプリケーション命令コードの先頭アドレスを取得し、取得したアドレスへ分岐する(ステップ1107、1110、1130)。この例では、図25のアプリケーションコードの先頭4000へ分岐する。CHCB(1,1)にアドレスが設定されていない場合、無効コマンド処理1200へ分岐する(ステップ1120)。

【0032】無効コマンド処理1200では、リターンコード311にエラーコードを設定し(ステップ1201)、コマンド終了処理1300へ分岐する(ステップ1202)。

【0033】共通コマンド処理3000あるいはアプリケーション専用コマンド処理4000が終了すると、コマンド終了1300に分岐してくる。ここでは、レスポンスデータ310に設定されているリターンコード311やコマンドリターンデータ312をホスト機器100に送信する(ステップ1301)。ここで再びコマンド受信241へ戻る。

【0034】図17に、システムコマンドデスパッチ処

理2000を示す。まずコマンドデータ300内からコマンド番号302を取込む(ステップ2001)。SCCB(360)をサーチし、コマンド番号302に対応するシステムコマンド処理ルーチンの先頭アドレスを取得し、取得したアドレスへ分岐する(ステップ2002)。この例では、Open_Channel、Close_Channelなどへ分岐する。各コマンド処理が終了したならば、コマンド終了1300へ分岐する(ステップ2009)。

【0035】図18はLoad_CCOMコマンド処理2300である。Load_CCOMコマンドは共通コマンドのロードユニット700をローディングする。共通コマンドのロードユニット700はコマンドデータ300内のコマンドパラメータ303に格納されている。

【0036】340のCCCBにおいて、CCCB(0)を登録状態にする(ステップ2301)。そして、共通コマンド命令コード702を共通コマンド動作領域263へ格納する。(ステップ2302)。次に共通コマンド用ワーク703のサイズを取得する(ステップ2303)。そして、340のCCCBデータを設定する。すなわち、CCCB(1)に共通コマンドコードへのポインタ、CCCB(2)に共通コマンド命令コード702のサイズ、CCCB(3)に共通コマンド用ワーク703のサイズ、CCCB(4)に共通コマンドの属性情報701を格納する(ステップ2304)。最後にリターンコード311に正常リターンコードを設定し、処理を終了する(ステップ2305、2360)。共通コマンド用のワーク領域は、後でアプリケーションがローディングされた時に割り振る。

【0037】図19はLoad_APコマンド処理2400である。Load_APコマンドはアプリケーションのロードユニット710をローディングする。アプリケーションのロードユニット710はコマンドデータ300内のコマンドパラメータ303に格納されている。

【0038】330のAPCBにおいて、空きエントリを探す。空きAPCBエントリを見つけたならば当該エントリのAPCB(1,0)を使用中状態にする(ステップ2401、2402)。アプリケーション命令コード712とアプリケーションデータS(713)を、アプリケーション動作領域510に格納する。アプリケーション動作領域510に別のアプリケーションが既に格納されている場合は、それに続いた動作領域520に格納する(ステップ2403)。次にアプリケーションデータD(714)を、アプリケーション用ワーク領域610に格納する。アプリケーション用ワーク領域610に別のアプリケーションが既に格納されている場合は、それに続いた動作領域620に格納する(ステップ2404)。このとき、アプリケーションデータD(611、621)に続いて共通コマンド用ワーク領域(612、622)を割り振る。共通コマンド用ワークのサイズは、340のCCCB(3)から得られる。当然、共通コマンドが未登録の場合、共通コマンド用ワーク領域(612、622)の割

り振りはスキップする(ステップ2405)。そしてAPCBのエントリ331に管理データを設定する。APCB(1,1)にアプリケーション命令コード(511、521)へのポインタ値、APCB(1,2)にアプリケーションデータS(512、522)へのポインタ値、APCB(1,3)にアプリケーションデータD(611、621)へのポインタ値、APCB(1,4)にアプリケーションデータD(611、621)のサイズ、APCB(1,7)に共通コマンド用ワーク(612、622)へのポインタ値を設定する。および、APCB(1,5)にアプリケーション名、APCB(1,6)にアプリケーション属性情報711を格納する。(ステップ2406)。最後にリターンコード311に正常リターンコードを設定し、処理を終了する(ステップ2407、2460)。APCBの先頭にあるAPCB管理データ339にて、アプリケーション動作領域262の空きエリアおよびアプリケーション用ワーク領域252の空きエリアを管理する。

【0039】図20はOpen_Channelコマンド処理2100である。Open_Channelコマンドは論理チャンネルをオープンし、指定されたアプリケーションを対応付ける、そして当該チャンネル番号を返す。

【0040】まず指定されたアプリケーションが使用中のチャンネルが既にオープンされているか否かチェックする。具体的には、350のCHCBのエントリ(351、352)内のCHCB(1,0)のチャンネル使用中アプリケーション番号をチェックする。(ステップ2101)。既にオープンしていた場合エラー終了する。オープンしていない場合オープン処理を行う。330のAPCBをチェックし、指定されたアプリケーションが登録されているか否かチェックする。登録されていない場合エラー終了する。(ステップ2102)。次に空きCHCBエントリ(351、352)を探す。空きCHCBエントリを見つけたならば、当該エントリのCHCB(1,0)にアプリケーション番号を設定し当該エントリを占有する(ステップ2103、2104)。そしてCHCBの管理データを設定する。CHCB(1,1)にアプリケーション命令コードへのポインタ値、CHCB(1,2)にアプリケーションデータSへのポインタ値、CHCB(1,3)にアプリケーションデータDへのポインタ値、CHCB(1,4)に共通コマンド用ワークへのポインタ値を設定する。これらのCHCBへの設定値は、当該アプリケーションプログラムを管理するAPCBエントリ(331、332)から得られる。(ステップ2105)。そして、当該アプリケーションデータD(611、621)および共通コマンド用ワーク(612、622)を初期化する(ステップ2106、2107)。最後にオープンしたチャンネル番号をコマンドリターンデータ312、リターンコード正常をリターンコード311に設定し、処理を終了する(ステップ2108、2109、2160)。

【0041】図21はClose_Channelコマンド処理220

0である。Close_Channelコマンドは指定されたチャンネルをクローズする。

【0042】まず、コマンドパラメータ303からコマンドパラメータを取り出す。パラメータで指定されたチャンネルに対応するCHCBのエントリ(351、352、)内のCHCB(1,0)をチェックする(ステップ2201)。CHCBがオープン状態ならば、CHCB(1,0)をクリアしてCHCBの占有を解除する、すなわち該当チャンネルをクローズする(ステップ2202)。最後にリターンコード正常をリターンコード311に設定し、処理を終了する(ステップ2203、2260)。

【0043】図22、図23に共通コマンド処理を示す。共通コマンド処理プログラムは、プログラムの先頭に共通コマンドデスパッチ処理3000を持つ。共通コマンドデスパッチ処理を経由してそれぞれのコマンド処理プログラム(3100、3200)へ分岐する。

【0044】コマンド起動1100が設定したカレントCHCB(320)の値にしたがってCHCBエントリ(351)を選択する(ステップ3001)。そして、当該CHCBエントリ(351)の管理する共通コマンド用ワークへのポインタとアプリケーションデータSへのポインタを選択する(ステップ3002)。これ以降、共通コマンド処理プログラムは、発行元のアプリケーションに依存した情報を格納するデータS(512)をアクセスできるようにする。また、オープンしている論理チャンネルに依存した共通コマンド用ワーク(612)をアクセスできるようにする。

【0045】次にコマンドデータ300内からコマンド番号302を取り出す(ステップ3003)。そして共通コマンド番号に対応した共通コマンド処理プログラム(3100、3200)へ分岐する(ステップ3004)。各コマンド処理が終了したならば、コマンド終了1300へ分岐する(ステップ3010)。

【0046】図23は共通コマンド処理プログラムの例である。

【0047】ステップ3002にて、当該CHCBエントリ(351)の管理する共通コマンド用ワークへのポインタとアプリケーションデータSへのポインタを選択してある。したがって、共有コマンド処理プログラムはアプリケーションデータS内のデータ(512)、共有コマンド用ワーク内のデータ(612)を自由にアクセスでき、そのデータを使って演算処理を実行できる。(ステップ3101、3102、3103、3104)。そして、コマンドリターンデータ312にレスポンスデータを設定する(ステップ3105)。最後に、リターンコード311にリターンコードを設定してコマンド処理を終了する(ステップ3106、3160)。

【0048】図24はアプリケーションデータS(512)の構成例である。データ領域の先頭にディレクトリ512dを配置する。ディレクトリには、各データへの

10

20

30

40

50

ポインタを格納しておき、各データを簡単にアクセスできる構造とする。

【0049】図25、図26にアプリケーション専用コマンド処理を示す。アプリケーション専用コマンド処理プログラムは、プログラムの先頭にアプリケーション専用コマンドデスパッチ処理4000を持つ。アプリケーション専用コマンドデスパッチ処理を経由してそれぞれのコマンド処理プログラム（4100、4200）へ分岐する。

【0050】まず、コマンドデータ300内からコマンド番号302を取り出す（ステップ4001）。そして共通コマンド番号に対応した共通コマンド処理プログラム（4100、4200）へ分岐する（ステップ4002）。各コマンド処理が終了したならば、コマンド終了1300へ分岐する（ステップ4010）。

【0051】図26アプリケーション専用コマンド処理の例である。アプリケーションデータS（512）およびアプリケーションデータD（611）内のデータアドレスを取得し（ステップ4101、4102）、それらのデータを使って演算を実行と演算結果の保存を行う（ステップ4103）。

【0052】そして、コマンドリターンデータ312にレスポンスデータを設定する（ステップ4104）。最後に、リターンコード311にリターンコードを設定してコマンド処理を終了する（ステップ4105、4160）。

【0053】本発明の実施例によれば、マルチアプリケーション対応のセキュアカードにて、アクティブなアプリケーションを複数オープンしそれらを同時に実行できる。また、それらのアプリケーション間で共通コマンド処理を使える。さらに、アプリケーション間共有コマンドを登録削除できる。またアプリケーション専用のコマンド処理プログラムも登録削除できる。

【0054】

【発明の効果】本発明によれば、複数のアプリケーションを並列に実行することができるため、ユーザの利便性を向上するという効果を奏する。

【図面の簡単な説明】

【図1】本発明の一実施例の全体構成図

【図2】本発明に関わる統合セキュリティカードのシステムブロック図

【図3】統合セキュリティカードに発行されるコマンドタイプの説明図

【図4】統合セキュリティカードのメモリマップ図

【図5】システムコマンド用ワーク領域とシステム管理データ領域の詳細図

【図6】アプリケーション動作領域とアプリケーション用ワーク領域の詳細図

【図7】アプリケーション処理プログラムの管理テーブル図

【図8】論理チャネルの管理テーブル図

【図9】システムコマンドの管理テーブル図

【図10】システムコマンドの説明図

【図11】共通コマンドとアプリケーションのロードユニットの説明図

【図12】コマンドデータとレスポンスデータの説明図

【図13】統合セキュリティカードの初期化処理のフロー図

【図14】コマンド受信処理のフロー図

【図15】コマンド起動処理のフロー図

【図16】無効コマンド処理、コマンド終了処理のフロー図

【図17】システムコマンドデスパッチ処理のフロー図

【図18】Load_CCOMコマンド処理のフロー図

【図19】Load_APコマンド処理のフロー図

【図20】Open_Channelコマンド処理のフロー図

【図21】Close_Channelコマンド処理のフロー図

【図22】共通コマンドデスパッチ処理のフロー図

【図23】共通コマンド処理のフロー図

【図24】アプリケーションデータSの構成図

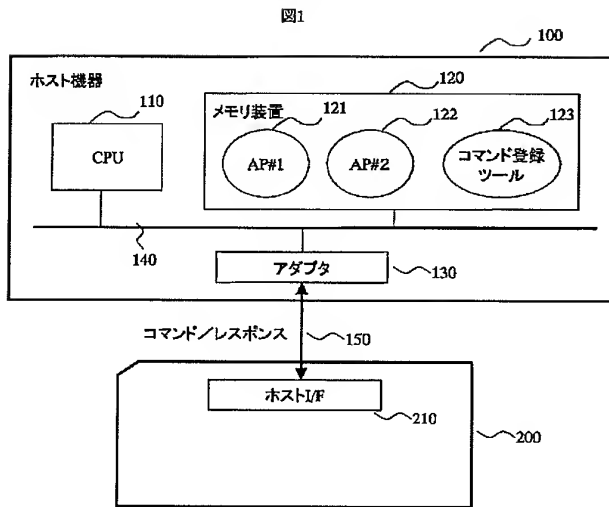
【図25】アプリケーション専用コマンドデスパッチ処理のフロー図

【図26】アプリケーション専用コマンド処理のフロー図

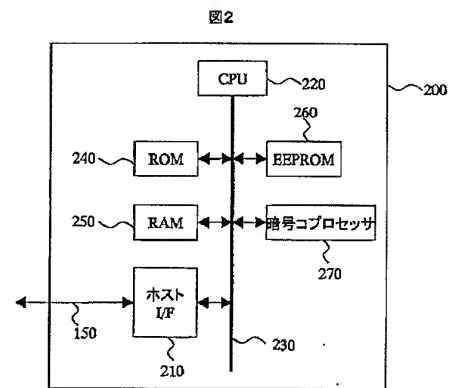
【符号の説明】

100…ホスト機器、110,220…中央処理装置、120…メモリ装置、121,122…アプリケーションプログラム、123…コマンド登録ツール、130…アダプタ、140…システムバス、150…インタフェースバス、200…総合セキュリティカード、210…ホストインタフェース、230…バス、240…ROM、250…RAM、260…EEPROM、270…暗号コプロセッサ

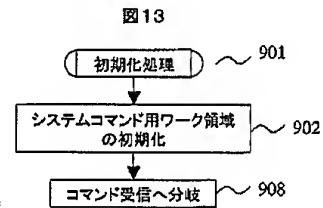
【図1】



【図2】



【図13】



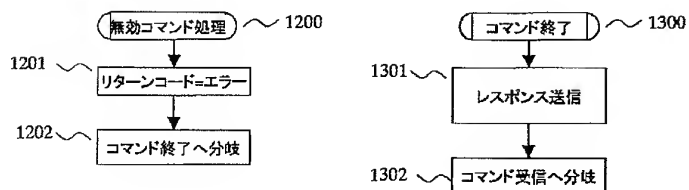
【図3】

図3

コマンドタイプ	処理内容
システムコマンド	システム制御用コマンド
共通コマンド	アプリケーション間で共通に使用できるコマンド (登録、削除可能)
アプリケーション 専用コマンド	アプリケーション専用コマンド (登録、削除可能)

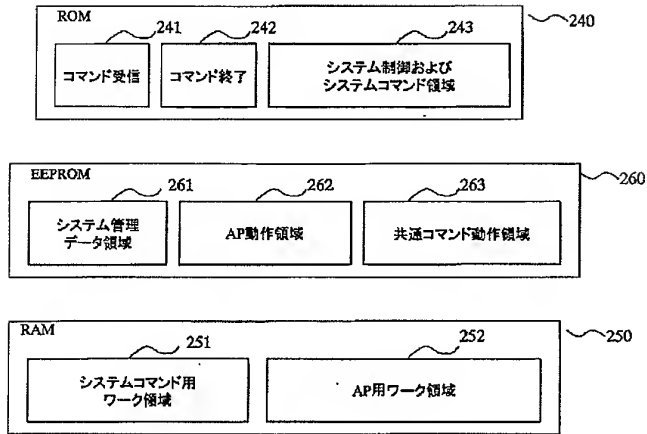
【図16】

図16



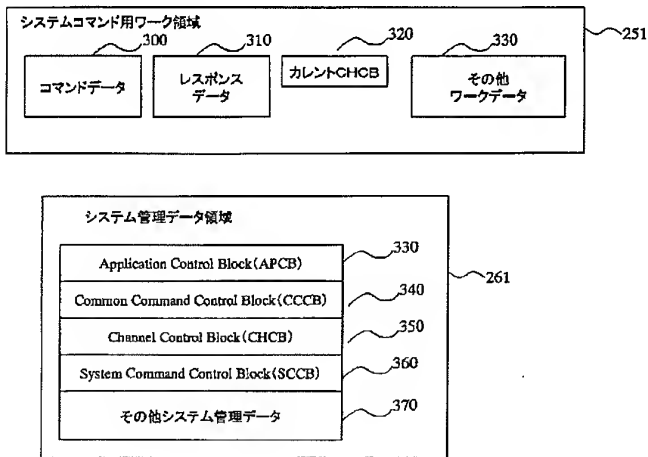
【図4】

図4



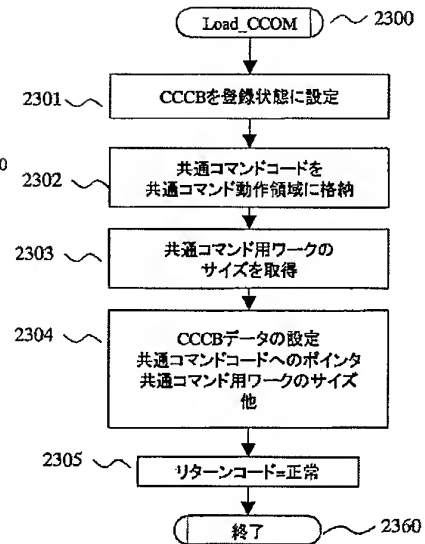
【図5】

図5



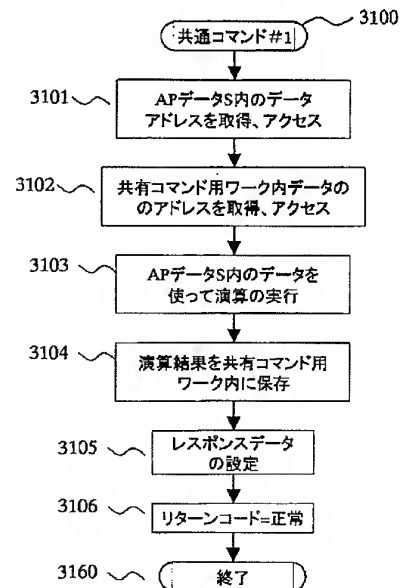
【図18】

図18



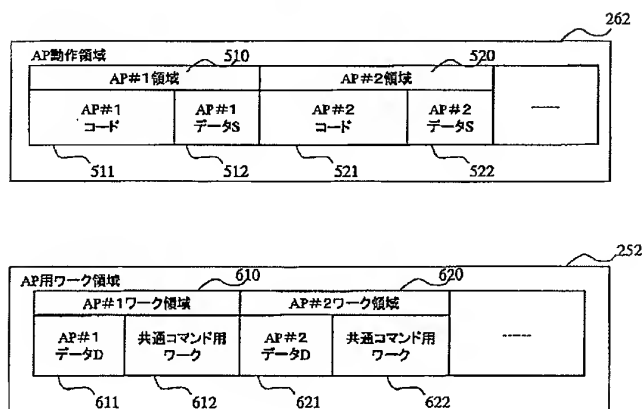
【図23】

図23



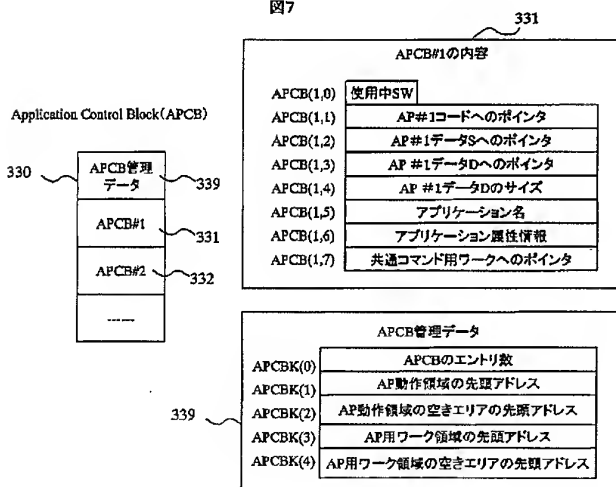
【図6】

図6



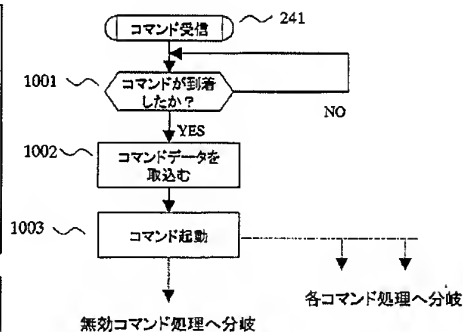
【図7】

図7



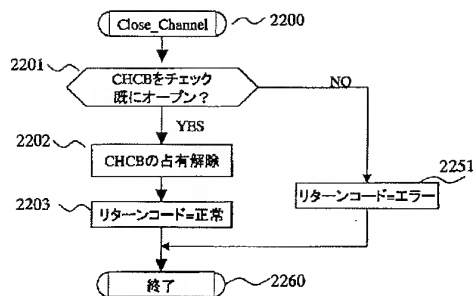
【図14】

図14



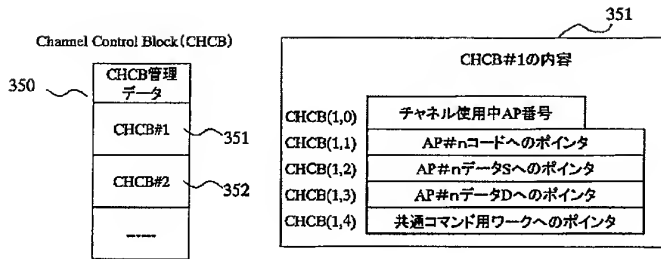
【図21】

図21



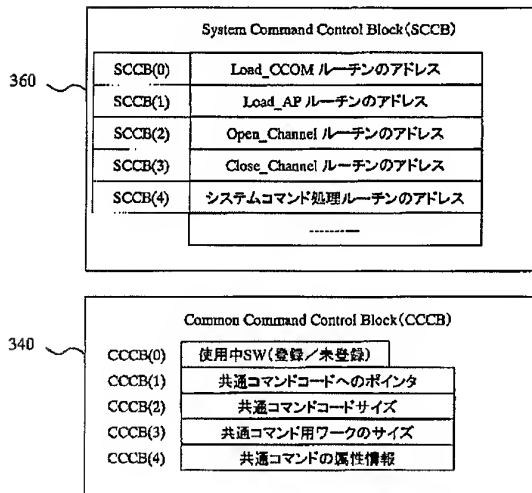
【図8】

図8



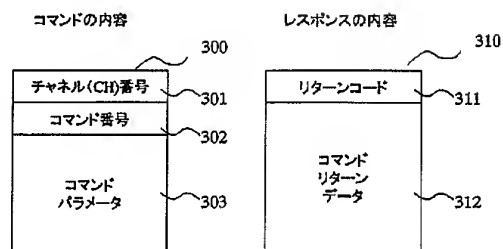
【図9】

図9



【図12】

図12

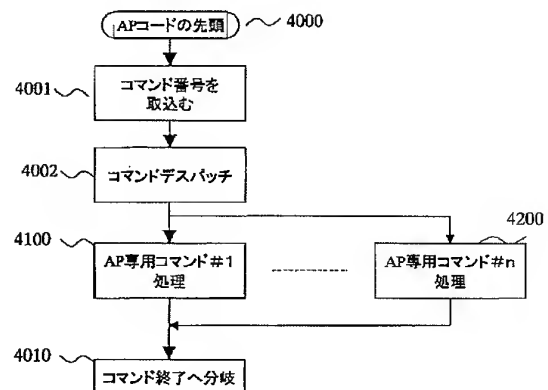


CH番号=0 : システムコマンド発行用チャネル
= 以外: ユーザ用チャネル

コマンド番号 00~1 = システムコマンド
I+1~J = 共通コマンド
J+1以上 = ユーザアプリ専用コマンド

【図25】

図25



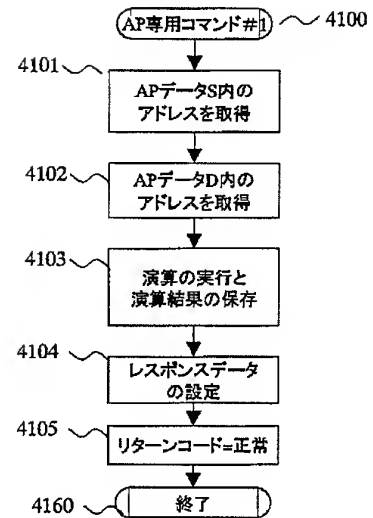
【図10】

図10

システムコマンド名	処理内容
Open_Channel	機能: チャンネルをオープンし、アプリケーションを対応づける 入力: CH番号=0 (システムコマンド) 対応づけるアプリケーション名 (番号) 出力: オープンしたチャンネル番号
Close_Channel	機能: チャンネルをクローズする 入力: CH番号=0 (システムコマンド) クローズするチャンネル番号
Load_CCOM	機能: 共通コマンドのローディング 入力: CH番号=0 (システムコマンド) 共通コマンドのロードユニット
Load_AP	機能: アプリケーションユニットのローディング 入力: CH番号=0 (システムコマンド) アプリケーションプログラムのロードユニット

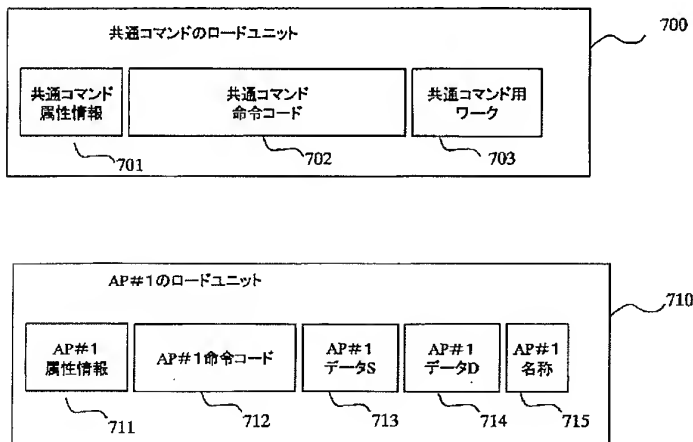
【図26】

図26

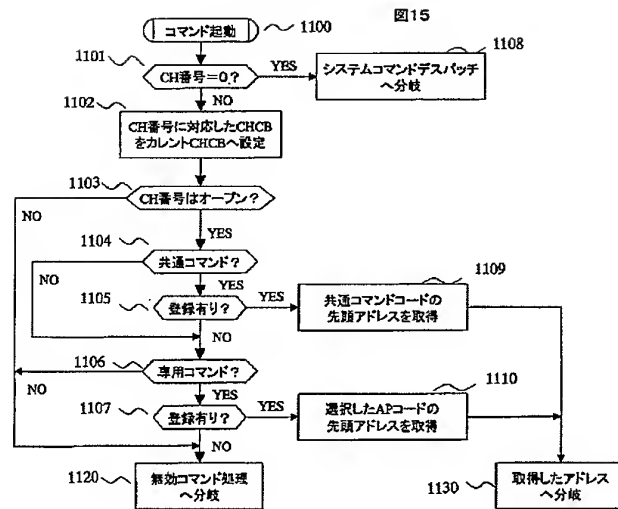


【図11】

図11

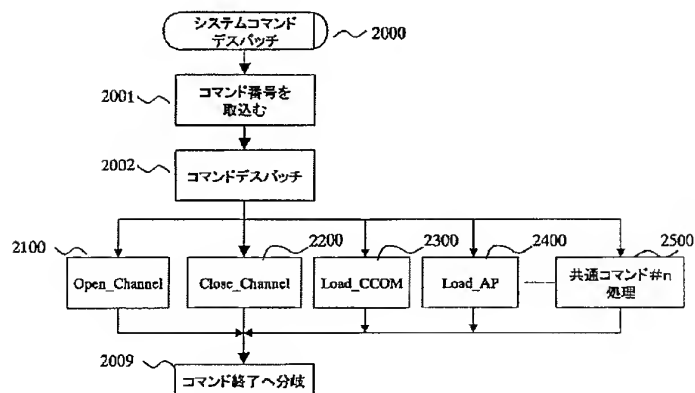


【図15】

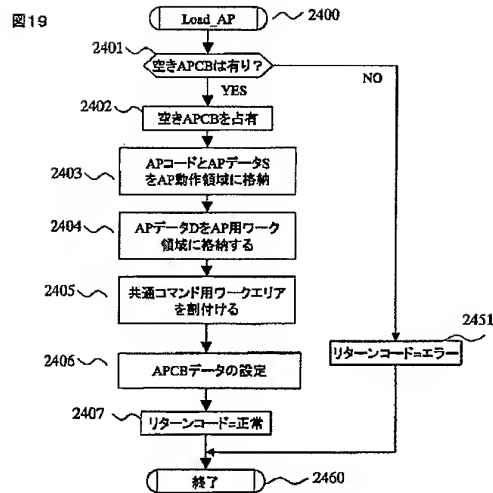


【図17】

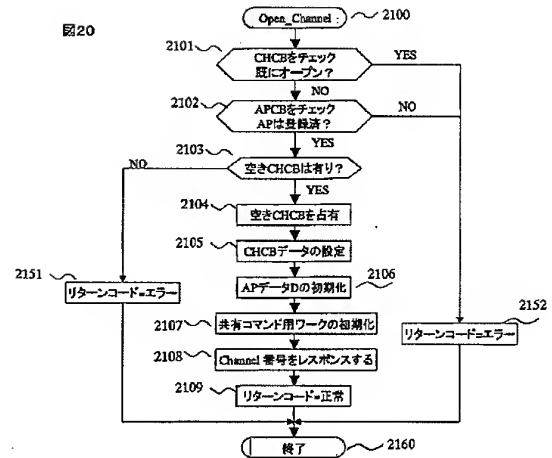
図17



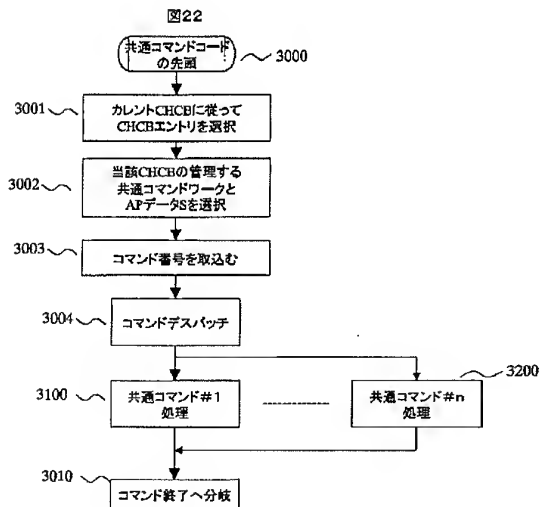
【図19】



【図20】

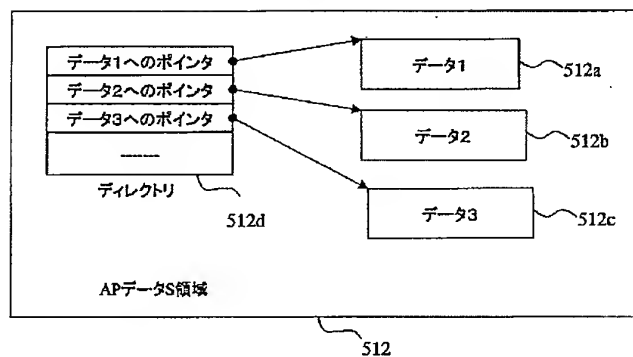


【図22】



【図24】

図24



フロントページの続き

(72)発明者 片山 国弘
 東京都小平市上水本町5丁目20番1号 株
 式会社日立製作所半導体グループ内
 (72)発明者 水島 永雅
 神奈川県川崎市麻生区王禅寺1099番地 株
 式会社日立製作所システム開発研究所内

Fターム(参考) 2C005 MA40 MB01 MB03 MB10 NA02
 NA08 SA02 SA05 SA06 SA08
 SA13 SA23 SA25 TA21 TA22
 5B035 AA06 AA13 BB09 CA38
 5B058 CA23 CA27 KA35
 5B076 DD05 FA00

CLAIMS

[Claim(s)]

[Claim 1]A security card provided with a registration deletion command of a common command processing program which can be used in common among two or more applications in a security card in which processing and memory of data are possible according to a command from host equipment.

[Claim 2]The security card according to claim 1 provided with a registration deletion command of an application device-dependent-command processing program which can be used with one application.

[Claim 3]A data division of an application program registered with a registration deletion command of said application device-dependent-command processing program, The security card according to claim 2 which can be used as a command parameter of a common command registered with a registration deletion command of said common command processing program.

[Claim 4]The security card according to claim 2 provided with a means to match and manage said application device-dependent-command processing program and said share command processing program, per logical channel.

[Claim 5]When carrying out loading of said common command processing program, compute a pointer to a common command instruction code, and size of a work area for common commands, and memory storage is made to memorize, The security card according to claim 2 provided with a means to assign a work area for common commands to said memory storage by application units when carrying out loading of said application device-dependent-command processing program.

[Claim 6]The security card according to claim 2 provided with a means to initialize dynamic data for application device dependent commands and work data for share commands corresponding to a case where a logical channel is opened.

[Claim 7]The security card according to claim 2 provided with a means which chooses and switches a work area for common commands by a logical channel number, and a means which chooses and switches static data for applications by a logical channel number.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention] This invention carries a security function and a memory storage function, and relates to the host equipment which can insert processing / [in which processing and memory are possible] storage (for example, security card), and its processing/storage (for example, security card) of data according to the command from host equipment.

[0002]

[Description of the Prior Art] An IC card embeds IC (integrated circuit) chip into a plastic card board, and has an external terminal of an IC chip in the surface. There are a power supply terminal, a clock terminal, a data input/output terminal, etc. in the external terminal of an IC chip. The IC chip operates, when a contact carries out direct supply of a power supply or the driving clock from an external terminal. By transmitting and receiving an electrical signal among contacts, such as a terminal, through an external terminal, an IC card exchanges information with a contact. As a result of information exchange, an IC card performs sending out of a calculation result or memory information, and change of memory information. The IC card can have the function to perform security processings, such as security data protection and personal authentication, based on such operation specifications. In the system for which the security of extra sensitive information, such as credit settlement and banking, is needed, the IC card is used as a user device for identification.

[0003]

[Problem to be solved by the invention] An IC card stores security data in an inside, and all processings in connection with security data are performed inside an IC card. By doing so, it has prevented security data coming out of an IC card. The processing in connection with security data is constituted as a command processing program. From the exterior (host equipment) of an IC card, if a command is transmitted to an IC card, the command processing program corresponding to the command will be executed. And an executed result is replied to host equipment. The command (or command processing program) which does not restrict a command in one but is equipped two or more pieces in many cases has a close relation to mutual, the executed result of the front command is saved inside, and processing of a next command is performed using the data in many cases. According to the scenario of the command issue for which it opts beforehand, a command is published by the IC card from host equipment. Security system processing is carried out by repeating a series of command transmission / response reception.

[0004] MULTOS etc. are known as an OS for IC cards corresponding to multi-application (operating system). The command processing program for applications and security data are

managed as one, and can be registered and deleted at an IC card at application units. And the firewall is set up between applications and mutual interference is prevented.

[0005] In the above-mentioned conventional technology, since the firewall was set up between applications, the common command used in common between applications was not able to be registered. Using the data in application as a command parameter to a common command etc. was not completed. There was a problem that it was difficult only for one to be able to carry out open [of the active application], but to perform two or more applications simultaneously.

[0006] The purpose of this invention is to provide the security card which improved the user's convenience by performing two or more applications in parallel.

[0007]

[Means for solving problem] This invention is provided with an open channel-command and closing channel command, the loading command of a common command processing program, and the loading command of an application device-dependent-command processing program.

[0008] A means to execute a system command, a common command, and an application device dependent command, the table and logical channel unit which manage many logical channels -- an application device-dependent-command processing program (an instruction code.) a static data, dynamic data, and share command processing program (an instruction code.) A means to compute and memorize the pointer to a common command instruction code, and the size of the work area for common commands when carrying out loading of the means and common command processing program which match and manage work data, A means to assign the work area for common commands to application units when carrying out loading of the application device-dependent-command processing program, A means to initialize the dynamic data for application device dependent commands and the work data for share commands corresponding to the case where a logical channel is opened, It has a means which makes command processing program selection and is performed with a logical channel number and a command number, a means which chooses and switches the work area for common commands by a logical channel number, and a means which chooses and switches the static data for applications by a logical channel number.

[0009]

[Mode for carrying out the invention] The working example of this invention is described using Drawings.

[0010] The entire configuration figure of one working example of this invention is shown in drawing 1. The integrated security card 200 in connection with this invention is connected to the adapter 130 of host equipment via the host interface 210. The host equipment 100 comprises the central processing unit (following CPU) 110, the memory apparatus 120, and the adapter 130 which perform control by the whole computer, and they are connected by the system bath 140. The application programs 121 and 122 and the command registration tool

123 are stored in the memory apparatus 120. These programs access the integrated security card 200. The command published from the programs 121, 122, and 123 is transmitted to the integrated security card 200 via the adapter 130 and the interface bus 150. From the integrated security card 200, a response is replied to the host equipment 100 corresponding to a command.

[0011]Drawing 2 is a system block figure of the integrated security card 200 in connection with this invention.

[0012]CPU220 for the integrated security card 200 to perform data processing, Data (a program.) containing -- with ROM(Read Only Memory) 240 for memorizing, and RAM (Random Access Memory)250 and EEPROM (Electrically Erasable Programmable ROM)260. It has the host interface 210 for transmitting and receiving the code co-processor 270, and the exterior and data for performing processing carried out to a code/decoding in between, and they are connected by bus 230. It is possible to perform cipher processing with the code co-processor 270 according to the command from the host equipment 100. According to a program (software), CPU220 may perform cipher processing instead of the code co-processor 270 (hardware). The integrated security card 200 communicates with the host equipment 100, and performs security processing and memory processing according to the command from the host equipment 100. The communication between the integrated security card 200 and the host equipment 100 may be a cable like the interface bus 150, and may be infrared rays and radio like an electric wave. The form of the integrated security card 200 is not limited in the shape of a card type. The integrated security card 200 is an IC card, for example.

[0013]The type of a command published by the integrated security card 200 from the host equipment 100 is shown in drawing 3. There are three types of commands, a system command, the common command which can be used in common between applications, and an application device dependent command. A common command and an application device-dependent-command processing program can be registered from the host equipment 100.

[0014]Drawing 4 is a memory map of ROM240 in the integrated security card 200, and EEPROM260 and RAM250. The command reception program 241, the command epilogue program 242, system control, and the system command field 243 are assigned to ROM240. The system management data area 261, the application operation field 262, and the common command active region 263 are assigned to EEPROM260. The work region 251 for system commands and the work region 252 for applications are assigned to RAM250. The system management data area 261 and the work region 251 for system commands are used for the command reception program 241, the command epilogue program 242, and the system command processing program in the system command field 243.

[0015]Drawing 5 is details of the work region 251 for system commands, and the system management data area 261. The command-data field which stores the command information

which received 300 from the host equipment 100, and 310 are response data areas which store the response information conversely returned to host equipment. 320 is current CHCB which memorizes the logical channel number chosen now.

[0016]Various system management tables are stored in the system management data area 261. The table APCB where 330 manages the application device-dependent-command processing program registered (Application Control Block). The table CCCB where 340 manages the common command processing program registered (Commoncommand control block). The table CHCB (Channel control block) where 350 manages a logical channel, and 360 are the tables SCCB (System command control block) which manage a system command.

[0017]Drawing 6 is details of the application operation field 262 and the work region 252 for applications. The application operation field 262 in EEPROM260 is divided into application units (510, 520), and the code (511, 521) and the data S (512, 522) of each application device-dependent-command processing program are stored in each field. The data S (512, 522) is static data of a constant etc. The work region 252 for applications in RAM250 is also divided into application units (610, 620), The data D (611, 621) and the work for common commands (612, 622) of each application device-dependent-command processing program are stored in each field. The data D (611, 621) is dynamic data used while application is active. The work for common commands (612, 622) is assigned for every application.

[0018]Drawing 7 is details of the table APCB (330) which manages the application device-dependent-command processing program ***** (ed). APCB (330) has an entry (331, 332, ..) corresponding to application. Each entry 331 stores the information about corresponding application. These information is created when application is registered.

[0019]APCB (1, 0) is a switch showing whether application is registered or not. There are a pointer to the work for common commands assigned by the place (pointer to a code, the data S, and the data D) where corresponding application is stored, an application name, and its application, etc. APCB (330) is created by the loading command (Load_AP) of the application unit shown in drawing 10.

[0020]Drawing 8 is details of the table CHCB (350) which manages a logical channel.

[0021]The program (121, 122, 123) in the host equipment 100 can access an application device dependent command in the integrated security card 200, and a common command via a logical channel. A logical channel is created by an open channel command (Open_Channel) shown in drawing 10, and a logical channel and application are matched. Corresponding to a logical channel number, it has an entry (351, 352, --). Each entry (351) stores information about a corresponding logical channel. CHCB (1, 0) expresses an application number while using the channel concerned. A pointer to a work for common commands assigned by a place (pointer to a code, the data S, and the data D) where application is stored during use, and its

application is stored.

[0022]Drawing 9 is details of the table CCCB (340) which manages the common command processing program which manages a system command, and which table-SCCB(s) (360) and is registered.

[0023]SCCB (360) stores the address of a system command manipulation routine. CCCB (340) stores the size of the register state of a common command, the pointer to common command code, and the work for common commands, etc. CCCB (340) is created by the loading command (Load_CCOM) of the common command shown in drawing 10.

[0024]The list of system commands is shown in drawing 10. An Open_Channel command opens a logical channel, and matches the specified application, and returns the channel designator concerned. A Close_Channel command closes the specified channel. A Load_CCOM command carries out loading of the common command processing program. A Load_AP command carries out loading of the application unit. When choosing a system command, the channel designator 0 is used.

[0025]The composition of the load unit 700 of the common command loaded by a Load_CCOM command and the composition of the load unit 710 of the application treated by a Load_AP command are shown in drawing 11. 700 comprises the common command attribution information 701, the common command instruction code 702, and the work 703 for common commands. 710 comprises the attribution information 711 of application, the application instruction code 712, the static data S (713), and the dynamic data D (714).

[0026]Drawing 12 is details of the command information stored in the command-data field 300, and details of the response information stored in the response data area 310. The command data 300 comprise the channel designator 301, the command number 302, and the command parameter 303. In system command issue, a channel designator is 0. It is distinguished with the command number 302 any of a system command, a common command, and a user application device dependent command they are. The response data 310 comprises the return code 311 and the command return data 312.

[0027]Drawing 13 shows the initialization processing 901 of the integrated security card 200. If the power supply of the integrated security card 200 is turned on, the initialization processing 901 will operate. The work region 251 for system commands is initialized (Step 902), and it branches to the command reception 241 (Step 908).

[0028]As for the command receiving process 241 and drawing 15, the command starting processing 1100 and drawing 16 of drawing 14 are the invalid command processing 1200 and the command end processing 1300. In the command receiving process 241, it is confirmed whether supervised the host interface 210 and the command arrived (Step 1001). Command data are incorporated into the command data 300 when a command arrives (Step 1002). And it branches to the command starting processing 1100 (Step 1003).

[0029]In the command starting processing 1100, the channel designator 301 is checked first (Step 1101). the case where a channel designator is 0 -- a system -- commun -- it branches to the DODESU patch 2000 (Step 1108).

[0030]When a channel designator is except zero, the number of the CHCB entry corresponding to a channel designator is set to current CHCB (320) (Step 1102). It controls about current CHCB henceforth. First, it is confirmed whether the channel concerned opens (Step 1103). When opened by the channel, the application number channel in use is set as CHCB (1, 0). When not opened, it branches to the invalid command processing 1200 (Step 1120). When opened, it is confirmed whether the command number 302 is a common command (Step 1104). Or it confirms whether be an application device dependent command (Step 1106). In the case of a common command, CCCB (340) is checked. If CCCB (0) is in a state with registration, the start address of common command code will be acquired from CCCB (1). And it branches to the acquired address (Steps 1105, 1109, and 1130). Here, it branches at the head of [3000] the common command code of drawing 22.

[0031]On the other hand, in the case of an application device dependent command, CHCB (1, 1) is checked. The start address of the application instruction code set as CHCB (1, 1) is acquired, and it branches to the acquired address (Steps 1107, 1110, and 1130). In this example, it branches at the head of [4000] the application codes of drawing 25. When the address is not set as CHCB (1, 1), it branches to the invalid command processing 1200 (Step 1120).

[0032]In the invalid command processing 1200, an error code is set as the return code 311 (Step 1201), and it branches to the command end processing 1300 (Step 1202).

[0033]After the common command processing 3000 or the application device-dependent-command processing 4000 is completed, it branches to the end 1300 of a command. Here, the return code 311 and the command return data 312 which are set as the response data 310 are transmitted to the host equipment 100 (Step 1301). It returns to the command reception 241 again here.

[0034]drawing 17 -- a system -- commun -- the DODESU patch processing 2000 is shown. The command number 302 is first incorporated from the inside of the command data 300 (Step 2001). SCCB (360) is searched, and the start address of the system command manipulation routine corresponding to the command number 302 is acquired, and it branches to the acquired address (Step 2002). In this example, it branches to Open_Channel, Close_Channel, etc. If each command processing is completed, it will branch to the end 1300 of a command (Step 2009).

[0035]Drawing 18 is the Load_CCOM command processing 2300. A Load_CCOM command carries out loading of the load unit 700 of a common command. The load unit 700 of the common command is stored in the command parameter 303 in the command data 300.

[0036]CCCB (0) is made into a register state in CCCB of 340 (Step 2301). And the common command instruction code 702 is stored in the common command active region 263. (Step 2302). Next, the size of the work 703 for common commands is acquired (Step 2303). And the CCCB data of 340 is set up. That is, the size of the work 703 for common commands is stored in the size of the common command instruction code 702, and CCCB (3), and the attribution information 701 of a common command is stored in CCCB (1) at CCCB (4) at the pointer to common command code, and CCCB (2) (Step 2304). Finally a normal return code is set as the return code 311, and processing is ended (Step 2305-2360). The work region for common commands is assigned when loading of the application is carried out later.

[0037]Drawing 19 is the Load_AP command processing 2400. A Load_AP command carries out loading of the load unit 710 of application. The load unit 710 of application is stored in the command parameter 303 in the command data 300.

[0038]A vacant entry is looked for in APCB of 330. If an empty APCB entry is found, APCB (1, 0) of the entry concerned will be made into busy status (Step 2401-2402). The application instruction code 712 and the application data S (713) are stored in the application operation field 510. When another application is already stored in the application operation field 510, it stores in the active region 520 which followed it (Step 2403). Next, the application data D (714) are stored in the work region 610 for applications. When another application is already stored in the work region 610 for applications, it stores in the active region 620 which followed it (Step 2404). At this time, the work region for common commands (612, 622) is assigned following the application data D (611, 621). The size of the work for common commands is obtained from CCCB (3) of 340. Naturally, when a common command is unregistered, assignment of the work region for common commands (612, 622) is skipped (Step 2405). And management data is set as the entry 331 of APCB. To APCB (1, 1), the pointer value to an application instruction code (511, 521), To APCB (1, 2), the pointer value to the application data S (512, 522), The pointer value to the work for common commands (612, 622) is set as APCB (1, 3) at the pointer value to the application data D (611,621), the size of the APCB (1, 4) application data D (611, 621), and APCB (1, 7). And the application attribution information 711 is stored in APCB (1, 5) at an application name and APCB (1, 6). (Step 2406). Finally a normal return code is set as the return code 311, and processing is ended (Steps 2407 and 2460). With the APCB management data 339 in the head of APCB, the empty area of the application operation field 262 and the empty area of the work region 252 for applications are managed.

[0039]Drawing 20 is the Open_Channel command processing 2100. An Open_Channel command opens a logical channel, and matches the specified application, and returns the channel designator concerned.

[0040]It confirms whether already opened by the channel which the application specified first is using. Specifically, an application number is checked during channel use of CHCB (1, 0) in the

entry (351,352) of CHCB of 350. (Step 2101). When already opened, the end of an error is carried out. When not opened, open shop operation is performed. It is confirmed whether check APCB of 330 and the specified application is registered. When not registered, the end of an error is carried out. (Step 2102). Next, it is vacant and a CHCB entry (351, 352) is looked for. If an empty CHCB entry is found, an application number is set as CHCB (1, 0) of the entry concerned, and the entry concerned is occupied (Steps 2103 and 2104). And the management data of CHCB is set up. The pointer value to the application data D is set as the pointer value to the application data S, and CHCB (1, 3), and the pointer value to the work for common commands is set as CHCB (1, 1) at CHCB (1, 4) at the pointer value to an application instruction code, and CHCB (1, 2). The preset value to these CHCB(s) is acquired from the APCB entry (331, 332) which manages the application program concerned. (Step 2105). And the application data D concerned (611, 621) and the work for common commands (612, 622) are initialized (Steps 2106 and 2107). The channel number opened at the end is set as the command return data 312, return code normalcy is set as the return code 311, and processing is ended (Steps 2108, 2109, and 2160).

[0041]Drawing 21 is the Close_Channel command processing 2200. A Close_Channel command closes a specified channel.

[0042]First, a command parameter is taken out from the command parameter 303. CHCB (1, 0) in an entry (351,352) of CHCB corresponding to a channel specified with a parameter is checked (Step 2201). If CHCB is an open condition, CHCB (1, 0) is cleared, and occupancy of CHCB is canceled, namely, an applicable channel is closed (Step 2202). Finally return code normalcy is set as the return code 311, and processing is ended (Steps 2203 and 2260).

[0043]Common command processing is shown in drawing 22 and drawing 23. a common command processing program is common to a head of a program -- commun -- it has the DODESU patch processing 3000. common -- commun -- it branches to each command processing program (3100, 3200) via DODESU patch processing.

[0044]A CHCB entry (351) is chosen according to the value of current CHCB (320) which the command starting 1100 set up (Step 3001). And the pointer to the work for common commands which the CHCB entry (351) concerned manages, and the pointer to the application data S are chosen (Step 3002). The common command processing program can access now after this the data S (512) which stores the information depending on the application of the publishing agency. The work (612) for common commands depending on the opened logical channel can be accessed now.

[0045]Next, the command number 302 is taken out from the inside of the command data 300 (Step 3003). And it branches to the common command processing program (3100-3200) corresponding to a common command number (Step 3004). If each command processing is completed, it will branch to the end 1300 of a command (Step 3010).

[0046]Drawing 23 is an example of a common command processing program.

[0047]The pointer to the work for common commands which the CHCB entry (351) concerned manages at Step 3002, and the pointer to the application data S are chosen. Therefore, the share command processing program can access freely the data (512) in the application data S, and the data (612) in the work for share commands, and can perform data processing using the data. (Steps 3101, 3102, 3103, and 3104). And response data is set as the command return data 312 (Step 3105). Finally, a return code is set as the return code 311, and command processing is ended (Step 3106-3160).

[0048]Drawing 24 is an example of composition of the application data S (512). The directory 512d is arranged at the head of a data area. The pointer to each data is stored in the directory, and it is considered as the structure which can access each data easily.

[0049]Application device-dependent-command processing is shown in drawing 25 and drawing 26. an application device-dependent-command processing program -- the head of a program -- only for application -- commun -- it has the DODESU patch processing 4000. only for application -- commun -- it branches to each command processing program (4100, 4200) via DODESU patch processing.

[0050]First, the command number 302 is taken out from the inside of the command data 300 (Step 4001). And it branches to the common command processing program (4100-4200) corresponding to a common command number (Step 4002). If each command processing is completed, it will branch to the end 1300 of a command (Step 4010).

[0051]It is an example of the drawing 26 application device-dependent-command processing. The data address in the application data S (512) and the application data D (611) is acquired (Steps 4101 and 4102), and preservation of execution and the result of an operation is performed for an operation using those data (Step 4103).

[0052]And response data is set as the command return data 312 (Step 4104). Finally, a return code is set as the return code 311, and command processing is ended (Step 4105-4160).

[0053]According to the working example of this invention, with the secure card corresponding to multi-application, two or more active applications are opened and they can be performed simultaneously. Common command processing can be used among those applications. The registration deletion of the application share command can be carried out. The registration deletion also of the command processing program only for application can be carried out.

[0054]

[Effect of the Invention]According to this invention, since two or more applications can be performed in parallel, the effect of improving a user's convenience is done so.

TECHNICAL FIELD

[Field of the Invention] This invention carries a security function and a memory storage function, and relates to the host equipment which can insert processing / [in which processing and memory are possible] storage (for example, security card), and its processing/storage (for example, security card) of data according to the command from host equipment.

PRIOR ART

[Description of the Prior Art]An IC card embeds IC (integrated circuit) chip into a plastic card board, and has an external terminal of an IC chip in the surface. There are a power supply terminal, a clock terminal, a data input/output terminal, etc. in the external terminal of an IC chip. The IC chip operates, when a contact carries out direct supply of a power supply or the driving clock from an external terminal. By transmitting and receiving an electrical signal among contacts, such as a terminal, through an external terminal, an IC card exchanges information with a contact. As a result of information exchange, an IC card performs sending out of a calculation result or memory information, and change of memory information. The IC card can have the function to perform security processings, such as security data protection and personal authentication, based on such operation specifications. In the system for which the security of extra sensitive information, such as credit settlement and banking, is needed, the IC card is used as a user device for identification.

EFFECT OF THE INVENTION

[Effect of the Invention]According to this invention, since two or more applications can be performed in parallel, the effect of improving a user's convenience is done so.

TECHNICAL PROBLEM

[Problem to be solved by the invention]An IC card stores security data in an inside, and all processings in connection with security data are performed inside an IC card. By doing so, it has prevented security data coming out of an IC card. Processing in connection with security data is constituted as a command processing program. From the exterior (host equipment) of an IC card, if a command is transmitted to an IC card, a command processing program corresponding to the command will be executed. And an executed result is replied to host equipment. A command (or command processing program) which does not restrict a command in one but is equipped two or more pieces in many cases has a close relation to mutual, an executed result of a front command is saved inside, and processing of a next command is performed using the data in many cases. According to a scenario of command issue for which it opts beforehand, a command is published by IC card from host equipment. Security system processing is carried out by repeating a series of command transmission / response reception. [0004]MULTOS etc. are known as an OS for IC cards corresponding to multi-application (operating system). A command processing program for applications and security data are managed as one, and can be registered and deleted at an IC card at application units. And a firewall is set up between applications and mutual interference is prevented.

[0005]In the above-mentioned conventional technology, since a firewall was set up between applications, a common command used in common between applications was not able to be registered. Using data in application as a command parameter to a common command etc. was not completed. There was a problem that it was difficult only for one to be able to carry out open [of the active application], but to perform two or more applications simultaneously.

[0006]The purpose of this invention is to provide a security card which improved a user's convenience by performing two or more applications in parallel.

MEANS

[Means for solving problem] This invention is provided with an open channel-command and closing channel command, a loading command of a common command processing program, and a loading command of an application device-dependent-command processing program.

[0008] A means to execute a system command, a common command, and an application device dependent command, a table and a logical channel unit which manage many logical channels -- an application device-dependent-command processing program (an instruction code.) a static data, dynamic data, and share command processing program (an instruction code.) A means to compute and memorize a pointer to a common command instruction code, and size of a work area for common commands when carrying out loading of a means and a common command processing program which match and manage work data, A means to assign a work area for common commands to application units when carrying out loading of the application device-dependent-command processing program, A means to initialize dynamic data for application device dependent commands and work data for share commands corresponding to a case where a logical channel is opened, It has a means which makes command processing program selection and is performed with a logical channel number and a command number, a means which chooses and switches a work area for common commands by a logical channel number, and a means which chooses and switches static data for applications by a logical channel number.

[0009]

[Mode for carrying out the invention] An working example of this invention is described using Drawings.

[0010] An entire configuration figure of one working example of this invention is shown in drawing 1. The integrated security card 200 in connection with this invention is connected to the adapter 130 of host equipment via the host interface 210. The host equipment 100 comprises the central processing unit (following CPU) 110, the memory apparatus 120, and the adapter 130 which perform control by the whole computer, and they are connected by the system bath 140. The application programs 121 and 122 and the command registration tool 123 are stored in the memory apparatus 120. These programs access the integrated security card 200. A command published from the programs 121, 122, and 123 is transmitted to the integrated security card 200 via the adapter 130 and the interface bus 150. From the integrated security card 200, a response is replied to the host equipment 100 corresponding to a command.

[0011] Drawing 2 is a system block figure of the integrated security card 200 in connection with this invention.

[0012] CPU220 for the integrated security card 200 to perform data processing, Data (a

program.) containing -- with ROM(Read Only Memory) 240 for memorizing, and RAM (Random Access Memory)250 and EEPROM (Electrically Erasable Programmable ROM)260. It has the host interface 210 for transmitting and receiving the code co-processor 270, and the exterior and data for performing processing carried out to a code/decoding in between, and they are connected by bus 230. It is possible to perform cipher processing with the code co-processor 270 according to a command from the host equipment 100. According to a program (software), CPU220 may perform cipher processing instead of the code co-processor 270 (hardware). The integrated security card 200 communicates with the host equipment 100, and performs security processing and memory processing according to a command from the host equipment 100. Communication between the integrated security card 200 and the host equipment 100 may be a cable like the interface bus 150, and may be infrared rays and radio like an electric wave. Form of the integrated security card 200 is not limited in the shape of a card type. The integrated security card 200 is an IC card, for example.

[0013]The type of a command published by the integrated security card 200 from the host equipment 100 is shown in drawing 3. There are three types of commands, a system command, the common command which can be used in common between applications, and an application device dependent command. A common command and an application device-dependent-command processing program can be registered from the host equipment 100.

[0014]Drawing 4 is a memory map of ROM240 in the integrated security card 200, and EEPROM260 and RAM250. The command reception program 241, the command epilogue program 242, system control, and the system command field 243 are assigned to ROM240. The system management data area 261, the application operation field 262, and the common command active region 263 are assigned to EEPROM260. The work region 251 for system commands and the work region 252 for applications are assigned to RAM250. The system management data area 261 and the work region 251 for system commands are used for the command reception program 241, the command epilogue program 242, and the system command processing program in the system command field 243.

[0015]Drawing 5 is details of the work region 251 for system commands, and the system management data area 261. A command-data field which stores command information which received 300 from the host equipment 100, and 310 are response data areas which store response information conversely returned to host equipment. 320 is current CHCB which memorizes a logical channel number chosen now.

[0016]Various system management tables are stored in the system management data area 261. The table APCB where 330 manages an application device-dependent-command processing program registered (Application Control Block). The table CCCB where 340 manages a common command processing program registered (Commoncommand control block). The table CHCB (Channel control block) where 350 manages a logical channel, and

360 are the tables SCCB (System command control block) which manage a system command.

[0017]Drawing 6 is details of the application operation field 262 and the work region 252 for applications. The application operation field 262 in EEPROM260 is divided into application units (510, 520), and a code (511, 521) and the data S (512, 522) of each application device-dependent-command processing program are stored in each field. The data S (512, 522) is static data of a constant etc. The work region 252 for applications in RAM250 is also divided into application units (610, 620), The data D (611, 621) and a work for common commands (612, 622) of each application device-dependent-command processing program are stored in each field. The data D (611, 621) is dynamic data used while application is active. A work for common commands (612, 622) is assigned for every application.

[0018]Drawing 7 is details of the table APCB (330) which manages the application device-dependent-command processing program ***** (ed). APCB (330) has an entry (331, 332, ..) corresponding to application. Each entry 331 stores the information about corresponding application. These information is created when application is registered.

[0019]APCB (1, 0) is a switch showing whether application is registered or not. There are a pointer to the work for common commands assigned by the place (pointer to a code, the data S, and the data D) where corresponding application is stored, an application name, and its application, etc. APCB (330) is created by the loading command (Load_AP) of the application unit shown in drawing 10.

[0020]Drawing 8 is details of the table CHCB (350) which manages a logical channel.

[0021]The program (121, 122, 123) in the host equipment 100 can access the application device dependent command in the integrated security card 200, and a common command via a logical channel. A logical channel is created by the open channel command (Open_Channel) shown in drawing 10, and a logical channel and application are matched. Corresponding to a logical channel number, it has an entry (351, 352, --). Each entry (351) stores the information about a corresponding logical channel. CHCB (1, 0) expresses an application number while using the channel concerned. The pointer to the work for common commands assigned by the place (pointer to a code, the data S, and the data D) where application is stored during use, and its application is stored.

[0022]Drawing 9 is details of the table CCCB (340) which manages the common command processing program which manages a system command, and which table-SCCB(s) (360) and is registered.

[0023]SCCB (360) stores the address of a system command manipulation routine. CCCB (340) stores the size of the register state of a common command, the pointer to common command code, and the work for common commands, etc. CCCB (340) is created by the loading command (Load_CCOM) of the common command shown in drawing 10.

[0024]The list of system commands is shown in drawing 10. An Open_Channel command opens a logical channel, and matches the specified application, and returns the channel designator concerned. A Close_Channel command closes the specified channel. A Load_CCOM command carries out loading of the common command processing program. A Load_AP command carries out loading of the application unit. When choosing a system command, the channel designator 0 is used.

[0025]Composition of the load unit 700 of a common command loaded by a Load_CCOM command and composition of the load unit 710 of application treated by a Load_AP command are shown in drawing 11. 700 comprises the common command attribution information 701, the common command instruction code 702, and the work 703 for common commands. 710 comprises the attribution information 711 of application, the application instruction code 712, the static data S (713), and the dynamic data D (714).

[0026]Drawing 12 is details of command information stored in the command-data field 300, and details of response information stored in the response data area 310. The command data 300 comprise the channel designator 301, the command number 302, and the command parameter 303. In system command issue, a channel designator is 0. It is distinguished with the command number 302 any of a system command, a common command, and a user application device dependent command they are. The response data 310 comprises the return code 311 and the command return data 312.

[0027]Drawing 13 shows the initialization processing 901 of the integrated security card 200. If the power supply of the integrated security card 200 is turned on, the initialization processing 901 will operate. The work region 251 for system commands is initialized (Step 902), and it branches to the command reception 241 (Step 908).

[0028]As for the command receiving process 241 and drawing 15, the command starting processing 1100 and drawing 16 of drawing 14 are the invalid command processing 1200 and the command end processing 1300. In the command receiving process 241, it is confirmed whether supervised the host interface 210 and the command arrived (Step 1001). Command data are incorporated into the command data 300 when a command arrives (Step 1002). And it branches to the command starting processing 1100 (Step 1003).

[0029]In the command starting processing 1100, the channel designator 301 is checked first (Step 1101). a case where a channel designator is 0 -- a system -- commun -- it branches to the DODESU patch 2000 (Step 1108).

[0030]When a channel designator is except zero, a number of a CHCB entry corresponding to a channel designator is set to current CHCB (320) (Step 1102). It controls about current CHCB henceforth. First, it is confirmed whether the channel concerned opens (Step 1103). When opened by channel, an application number channel in use is set as CHCB (1, 0). When not opened, it branches to the invalid command processing 1200 (Step 1120). When opened, it is

confirmed whether the command number 302 is a common command (Step 1104). Or it confirms whether be an application device dependent command (Step 1106). In the case of a common command, CCCB (340) is checked. If CCCB (0) is in a state with registration, a start address of common command code will be acquired from CCCB (1). And it branches to an acquired address (Steps 1105, 1109, and 1130). Here, it branches at the head of [3000] common command code of drawing 22.

[0031]On the other hand, in the case of an application device dependent command, CHCB (1, 1) is checked. The start address of the application instruction code set as CHCB (1, 1) is acquired, and it branches to the acquired address (Steps 1107, 1110, and 1130). In this example, it branches at the head of [4000] the application codes of drawing 25. When the address is not set as CHCB (1, 1), it branches to the invalid command processing 1200 (Step 1120).

[0032]In the invalid command processing 1200, an error code is set as the return code 311 (Step 1201), and it branches to the command end processing 1300 (Step 1202).

[0033]After the common command processing 3000 or the application device-dependent-command processing 4000 is completed, it branches to the end 1300 of a command. Here, the return code 311 and the command return data 312 which are set as the response data 310 are transmitted to the host equipment 100 (Step 1301). It returns to the command reception 241 again here.

[0034]drawing 17 -- a system -- commun -- the DODESU patch processing 2000 is shown. The command number 302 is first incorporated from the inside of the command data 300 (Step 2001). SCCB (360) is searched, and a start address of a system command manipulation routine corresponding to the command number 302 is acquired, and it branches to an acquired address (Step 2002). In this example, it branches to Open_Channel, Close_Channel, etc. If each command processing is completed, it will branch to the end 1300 of a command (Step 2009).

[0035]Drawing 18 is the Load_CCOM command processing 2300. A Load_CCOM command carries out loading of the load unit 700 of a common command. The load unit 700 of a common command is stored in the command parameter 303 in the command data 300.

[0036]CCCB (0) is made into a register state in CCCB of 340 (Step 2301). And the common command instruction code 702 is stored in the common command active region 263. (Step 2302). Next, size of the work 703 for common commands is acquired (Step 2303). And CCCB data of 340 is set up. That is, size of the work 703 for common commands is stored in size of the common command instruction code 702, and CCCB (3), and the attribution information 701 of a common command is stored in CCCB (1) at CCCB (4) at a pointer to common command code, and CCCB (2) (Step 2304). Finally a normal return code is set as the return code 311, and processing is ended (Step 2305-2360). A work region for common commands is assigned

when loading of the application is carried out later.

[0037]Drawing 19 is the Load_AP command processing 2400. A Load_AP command carries out loading of the load unit 710 of application. The load unit 710 of application is stored in the command parameter 303 in the command data 300.

[0038]A vacant entry is looked for in APCB of 330. If an empty APCB entry is found, APCB (1, 0) of the entry concerned will be made into busy status (Step 2401-2402). The application instruction code 712 and the application data S (713) are stored in the application operation field 510. When another application is already stored in the application operation field 510, it stores in the active region 520 which followed it (Step 2403). Next, the application data D (714) are stored in the work region 610 for applications. When another application is already stored in the work region 610 for applications, it stores in the active region 620 which followed it (Step 2404). At this time, the work region for common commands (612, 622) is assigned following the application data D (611, 621). The size of the work for common commands is obtained from CCCB (3) of 340. Naturally, when a common command is unregistered, assignment of the work region for common commands (612, 622) is skipped (Step 2405). And management data is set as the entry 331 of APCB. To APCB (1, 1), the pointer value to an application instruction code (511, 521), To APCB (1, 2), the pointer value to the application data S (512, 522), The pointer value to the work for common commands (612, 622) is set as APCB (1, 3) at the pointer value to the application data D (611,621), the size of the APCB (1, 4) application data D (611, 621), and APCB (1, 7). And the application attribution information 711 is stored in APCB (1, 5) at an application name and APCB (1, 6). (Step 2406). Finally a normal return code is set as the return code 311, and processing is ended (Steps 2407 and 2460). With the APCB management data 339 in the head of APCB, the empty area of the application operation field 262 and the empty area of the work region 252 for applications are managed.

[0039]Drawing 20 is the Open_Channel command processing 2100. An Open_Channel command opens a logical channel, and matches specified application, and returns the channel designator concerned.

[0040]It confirms whether already opened by channel which application specified first is using. Specifically, an application number is checked during channel use of CHCB (1, 0) in an entry (351,352) of CHCB of 350. (Step 2101). When already opened, the end of an error is carried out. When not opened, open shop operation is performed. It is confirmed whether check APCB of 330 and specified application is registered. When not registered, the end of an error is carried out. (Step 2102). Next, it is vacant and a CHCB entry (351, 352) is looked for. If an empty CHCB entry is found, an application number is set as CHCB (1, 0) of the entry concerned, and the entry concerned is occupied (Steps 2103 and 2104). And management data of CHCB is set up. A pointer value to the application data D is set as a pointer value to the application data S, and CHCB (1, 3), and a pointer value to a work for common commands

is set as CHCB (1, 1) at CHCB (1, 4) at a pointer value to an application instruction code, and CHCB (1, 2). A preset value to these CHCB(s) is acquired from an APCB entry (331, 332) which manages the application program concerned. (Step 2105). And the application data D concerned (611, 621) and a work for common commands (612, 622) are initialized (Steps 2106 and 2107). A channel number opened at the end is set as the command return data 312, return code normalcy is set as the return code 311, and processing is ended (Steps 2108, 2109, and 2160).

[0041]Drawing 21 is the Close_Channel command processing 2200. A Close_Channel command closes the specified channel.

[0042]First, a command parameter is taken out from the command parameter 303. CHCB (1, 0) in the entry (351,352) of CHCB corresponding to the channel specified with the parameter is checked (Step 2201). If CHCB is an open condition, CHCB (1, 0) is cleared, and occupancy of CHCB is canceled, namely, an applicable channel is closed (Step 2202). Finally return code normalcy is set as the return code 311, and processing is ended (Steps 2203 and 2260).

[0043]Common command processing is shown in drawing 22 and drawing 23. a common command processing program is common to the head of a program -- commun -- it has the DODESU patch processing 3000. common -- commun -- it branches to each command processing program (3100, 3200) via DODESU patch processing.

[0044]A CHCB entry (351) is chosen according to a value of current CHCB (320) which the command starting 1100 set up (Step 3001). And a pointer to a work for common commands which the CHCB entry (351) concerned manages, and a pointer to the application data S are chosen (Step 3002). The common command processing program can access now after this the data S (512) which stores information depending on application of a publishing agency. A work (612) for common commands depending on an opened logical channel can be accessed now.

[0045]Next, the command number 302 is taken out from the inside of the command data 300 (Step 3003). And it branches to a common command processing program (3100-3200) corresponding to a common command number (Step 3004). If each command processing is completed, it will branch to the end 1300 of a command (Step 3010).

[0046]Drawing 23 is an example of a common command processing program.

[0047]A pointer to a work for common commands which the CHCB entry (351) concerned manages at Step 3002, and a pointer to the application data S are chosen. Therefore, the share command processing program can access freely data (512) in the application data S, and data (612) in a work for share commands, and can perform data processing using the data. (Steps 3101, 3102, 3103, and 3104). And response data is set as the command return data 312 (Step 3105). Finally, a return code is set as the return code 311, and command processing is ended (Step 3106-3160).

[0048]Drawing 24 is an example of composition of the application data S (512). The directory 512d is arranged at the head of a data area. A pointer to each data is stored in a directory, and it is considered as structure which can access each data easily.

[0049]Application device-dependent-command processing is shown in drawing 25 and drawing 26. an application device-dependent-command processing program -- the head of a program -- only for application -- commun -- it has the DODESU patch processing 4000. only for application -- commun -- it branches to each command processing program (4100, 4200) via DODESU patch processing.

[0050]First, the command number 302 is taken out from the inside of the command data 300 (Step 4001). And it branches to the common command processing program (4100-4200) corresponding to a common command number (Step 4002). If each command processing is completed, it will branch to the end 1300 of a command (Step 4010).

[0051]It is an example of the drawing 26 application device-dependent-command processing. The data address in the application data S (512) and the application data D (611) is acquired (Steps 4101 and 4102), and preservation of execution and the result of an operation is performed for an operation using those data (Step 4103).

[0052]And response data is set as the command return data 312 (Step 4104). Finally, a return code is set as the return code 311, and command processing is ended (Step 4105-4160).

[0053]According to the working example of this invention, with a secure card corresponding to multi-application, two or more active applications are opened and they can be performed simultaneously. Common command processing can be used among those applications. The registration deletion of the application share command can be carried out. The registration deletion also of the command processing program only for application can be carried out.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1]The entire configuration figure of one working example of this invention

[Drawing 2]The system block figure of the integrated security card in connection with this invention

[Drawing 3]The explanatory view of the command type published by the integrated security card

[Drawing 4]The memory map figure of an integrated security card

[Drawing 5]The detail view of the work region for system commands, and a system management data area

[Drawing 6]The detail view of an application operation field and the work region for applications

[Drawing 7]The management table figure of an application process program

[Drawing 8]The management table figure of a logical channel

[Drawing 9]The management table figure of a system command

[Drawing 10]The explanatory view of a system command

[Drawing 11]The explanatory view of the load unit of a common command and application

[Drawing 12]The explanatory view of command data and response data

[Drawing 13]The flow chart of the initialization processing of an integrated security card

[Drawing 14]The flow chart of a command receiving process

[Drawing 15]The flow chart of command starting processing

[Drawing 16]The flow chart of invalid command processing and command end processing

[Drawing 17]a system -- commun -- the flow chart of DODESU patch processing

[Drawing 18]The flow chart of Load_CCOM command processing

[Drawing 19]The flow chart of Load_AP command processing

[Drawing 20]The flow chart of Open_Channel command processing

[Drawing 21]The flow chart of Close_Channel command processing

[Drawing 22]common -- commun -- the flow chart of DODESU patch processing

[Drawing 23]The flow chart of common command processing

[Drawing 24]The block diagram of the application data S

[Drawing 25]only for application -- commun -- the flow chart of DODESU patch processing

[Drawing 26]The flow chart of application device-dependent-command processing

[Explanations of letters or numerals]

100 -- Host equipment, 110,220 -- A central processing unit, 120 -- Memory apparatus,

121,122 -- An application program, 123 -- Command registration tool, 130 [-- A

comprehensive security card 210 / -- A host interface, 230 / -- A bus, 240 / -- ROM, 250 / --

RAM, 260 / -- EEPROM, 270 / -- Code co-processor] -- An adapter, 140 -- A system bath, 150
-- An interface bus, 200